Kobi Nisim & Itay Yaakov

More than 200,000 terrorist attacks worldwide, 1970-2019

Research questions:

- Is it possible to predict, in a particular month, in a particular country the number of people killed in terrorist attacks?
- What pattern can we observe and what conclusions can we learn from them? them? (according to the data)

Data Sources:

Crawling: Global Terrorism DATABASE

Managed by the National Consortium for the Study of Terrorism and Responses to Terrorism (START), the Global Terrorism Database™ includes more than 200,000 terrorist attacks dating back to 1970. Read about the history of the GTD below.

GTD
GLOBAL TERRORISM DATABASE

During the search for suitable material for our project, which contains as many terrorist attacks as possible from as many countries in the world, we found a perfect site, a site that contains a lot of valuable information. The site contains a huge amount of material, about terrorist attacks that have taken place around the world, arranged according to characteristics that exactly fit our needs. The site contains extensive information spread over a large number of years which can provide maximum results and an optimal answer to the forecasting question.

Use of libraries during the project

Imports

```
import json
import pandas as pd
import requests
import scipy as sc
import numpy as np
import matplotlib as mpl
import matplotlib.pyplot as plt
import matplotlib.cm as cm
import seaborn as sns
%matplotlib inline
#the graphs is inline
from time import sleep
#import sklearn func for regression
from sklearn import linear_model
from sklearn import ensemble
#import visualization setup
plt.rcParams['figure.figsize'] = (10, 6)
from mpl_toolkits.mplot3d import Axes3D
from matplotlib import cm
```

Execution process crawling:

- We used the 'BeautifulSoup' library to scan the relevant records from DATABASE.

```
from bs4 import BeautifulSoup
import time
from datetime import datetime
```

- We put the values from the scan we performed into a neat table of lists.

- Finally, everything is stored in a Data-Frame that displays the data in a concatenation of all the information into one table while adjusting the columns into a uniform format on which the study can be performed.

```
df=pd.DataFrame.from_dict({"DATE": DATE, "COUNTRY": COUNTRY,"CITY":CITY, "PERPETRATOR_GROUP":PERPETRATOR_GROUP,
                           "FATALITIES":FATALITIES, "INJURED":INJURED, "TARGET_TYPE":TARGET_TYPE,"YEAR":YEAR})
```

```
DATE=list()
COUNTRY=list()
CITY=list()
PERPETRATOR_GROUP=list()
FATALITIES=list()
INJURED=list()
TARGET_TYPE=list()
YEAR=list()

for n_page in range(1,880):
    url=f'https://www.start.umd.edu/gtd/search/Results.aspx
    html=requests.get(url)
```

```
if html.status_code != 200:
    print("the requst not succes ")
soup=BeautifulSoup(html.content,"html.parser")
tbl = soup("table",attrs={"class":"results"})[0]
Elements=tbl.find("tbody")
for element in Elements.find_all("tr"):
        Line=element.find_all("td")
        date=(Line[1].string)   #strint to time
        my_time = time.strptime(date, '%Y-%m-%d')
        timestamp = time.mktime(my_time)
        my_datetime = datetime.fromtimestamp(timestamp)
        DATE.append(my_datetime)
        YEAR.append(my_datetime.year)
        COUNTRY.append(Line[2].string)
        CITY.append(Line[3].string)
        PERPETRATOR_GROUP.append(Line[4].string)
        FATALITIES.append(Line[5].string)
        INJURED.append(Line[6].string)
        TARGET_TYPE.append(Line[7].string)
```

Data cleaning:

- Filter duplicates and empty values

```
def count_duplicatives(df):
    if(df.duplicated().sum() != 0):
        return remove_duplicatives(df)
```

```
def remove_duplicatives(df):
    return df.drop_duplicates(keep=False,inplace=True)
```

- Convert string values to int values (INJURED, FATALITIES)

```
def str_TO_int(df):
    df["INJURED"]=df["INJURED"].astype(int)
    df["FATALITIES"]=df["FATALITIES"].astype(int)
```

- Replace string value with 'UNKNOWN' to NaN

```
def Unknown_To_NaN(df):
    return df.replace('Unknown',np.nan)
```

- Removing NaN values

```
def Removing_NaN(df):
    return df.dropna()
```

Data cleaning:

Arranging the information in a table

After cleaning and arranging the large amount of raw information we were able to extract from the site, as mentioned, we entered all the data into an orderly table, free of 'junk' values, empty entries and after filtering duplicate entries. The table after cleaning contains a smaller number of records but those that provide quality and reliable information without repetitions or values that cannot be processed. The table now contains data that can be worked with and perform in-depth research that will provide us with the desired prediction result.

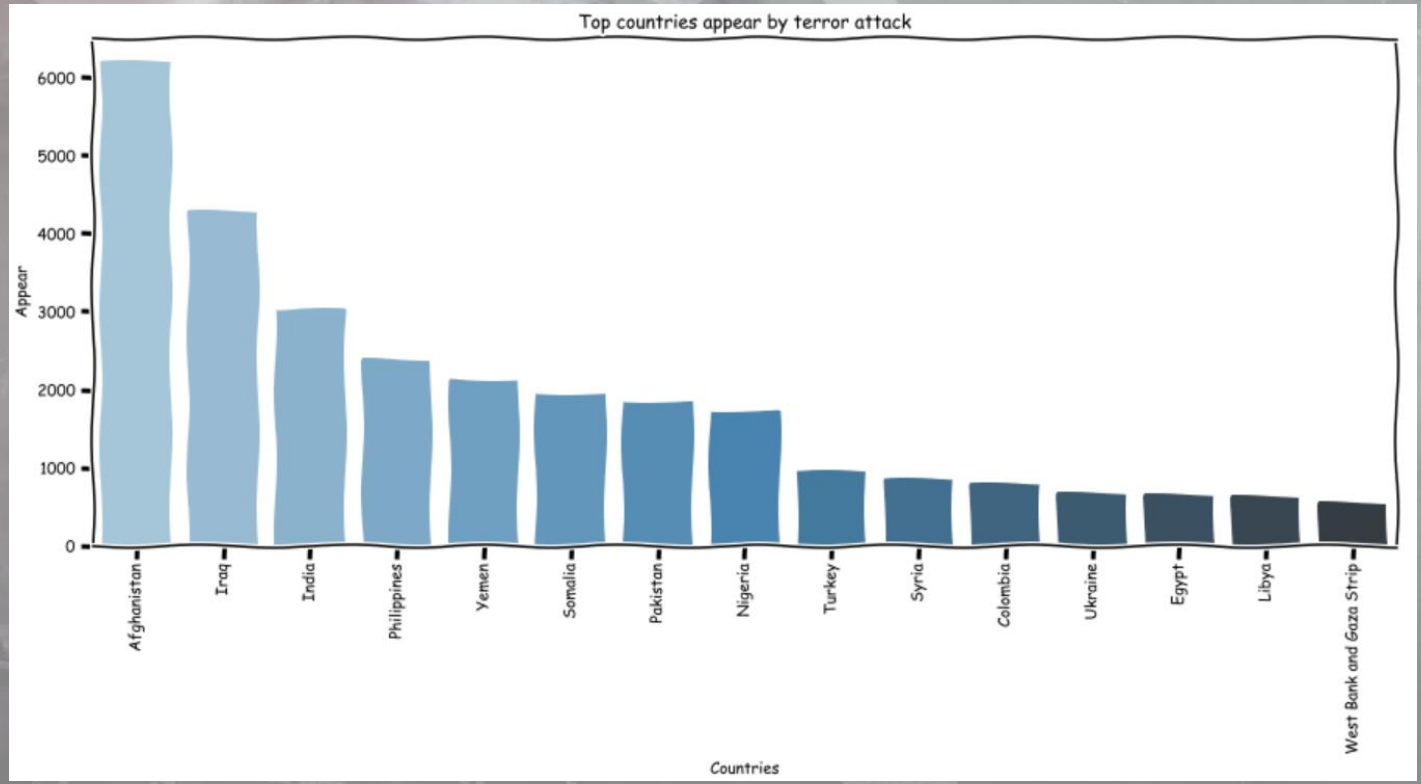
	Unnamed: 0	DATE	COUNTRY	CITY	PERPETRATOR_GROUP	FATALITIES	INJURED	TARGET_TYPE	YEAR
0	5	2019-12-30	Iraq	Nada	Islamic State of Iraq and the Levant (ISIL)	1	0	Military	2019
1	6	2019-12-31	Cameroon	Njap	Separatists	1	0	Private Citizens & Property	2019
2	9	2019-12-31	Russia	Magas	Caucasus Province of the Islamic State	2	4	Police	2019
3	10	2019-12-31	Philippines	Kinamaybay	New People's Army (NPA)	1	0	Private Citizens & Property	2019
4	12	2019-12-31	Syria	Tabqah	Islamic State of Iraq and the Levant (ISIL)	3	0	Private Citizens & Property	2019
...	...	...	...	...	...	...	...	...	...
34502	87894	2012-12-31	Pakistan	Jamrud	Muslim extremists	1	0	Educational Institution	2012
34503	87896	2012-12-31	Afghanistan	Khaki Safed	Taliban	1	3	Police	2012
34504	87897	2012-12-31	Iraq	Balad Ruz	Al-Qaida in Iraq	0	4	Private Citizens & Property	2012
34505	87898	2012-12-31	Iraq	Baghdad	Al-Qaida in Iraq	0	6	Private Citizens & Property	2012
34506	87899	2012-12-31	Iraq	Kirkuk	Al-Qaida in Iraq	2	0	Military	2012

EDA

Top countries with the most terror attack

First, we wanted to visually see the 15 countries with the most terrorist attacks in the entire course of history. We used a column graph configuration which shows on the one hand the amount of attacks that have taken place over the years and is arranged according to the names of the countries on the other.

```
plt.subplots(figsize=(15,6))
sns.barplot(all_data['COUNTRY'].value_counts()[:15].index,all_data['COUNTRY'].
plt.title('Top countries appear by terror attack')
plt.xlabel('Countries')
plt.ylabel('Appear')
plt.xticks(rotation= 90)#rotation to countries text
plt.show()
```



As a preparation for displaying appropriate graphs for displaying the desired data, variables have been introduced to help us display the desired information.

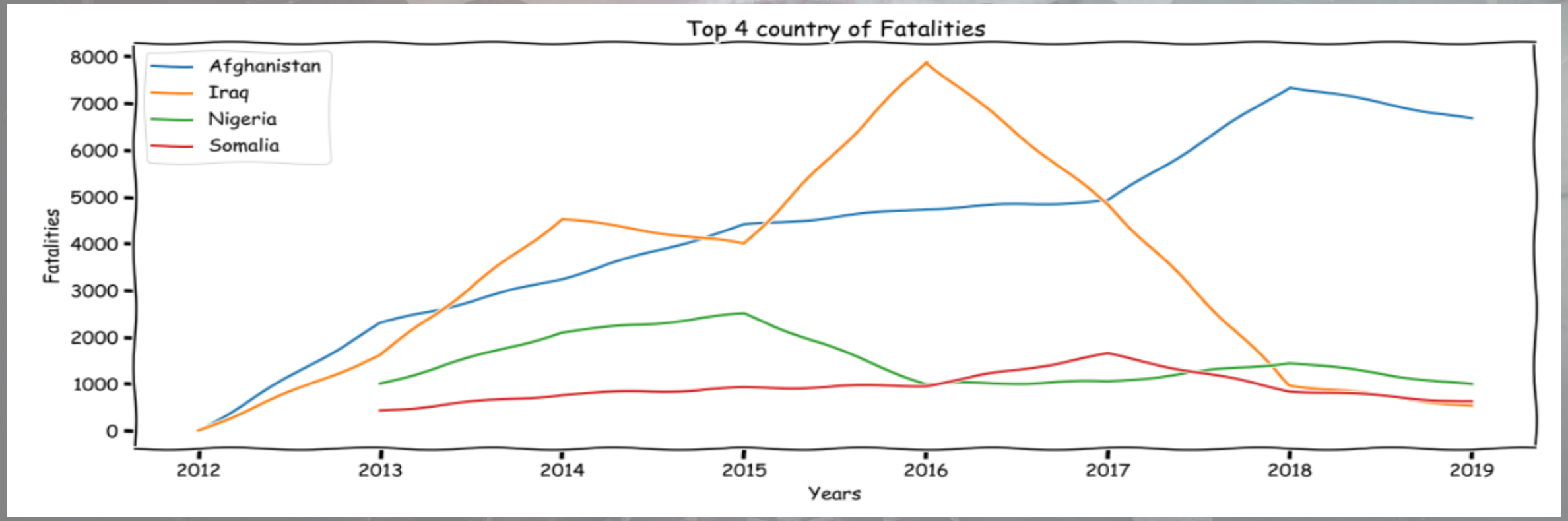
```
new=all_data.groupby(['COUNTRY']).sum()
TOP_INJURED=new.sort_values('INJURED').iloc[::-1][0:5]
TOP_FATALITIES=new.sort_values('FATALITIES').iloc[::-1][0:5]##revers
#TOP1=TOP.iloc[:,0]

DATA = all_data.groupby(['COUNTRY', 'YEAR']).sum()
print(DATA)


####top3-TOP_INJURED
index0 =DATA.loc[TOP_INJURED.index[0]]
index1 =DATA.loc[TOP_INJURED.index[1]]
index2 =DATA.loc[TOP_INJURED.index[2]]
index31 =DATA.loc[TOP_INJURED.index[3]]

##top3-TOP_FATALITIES
index3 =DATA.loc[TOP_FATALITIES.index[0]]
index4 =DATA.loc[TOP_FATALITIES.index[1]]
index5 =DATA.loc[TOP_FATALITIES.index[2]]
index6 =DATA.loc[TOP_FATALITIES.index[3]]
```

Question1:

Which four countries with most Fatalities?

You can learn from the graph who the four countries with the most deaths over the last decade are. As a direct result of the attacks that took place in their territory.

```
plt.xkcd()
plt.figure(figsize=[15,6])#size of the plt
plt.plot(YEAR3, FATALITIES0, label = TOP_FATALITIES.index[0])
plt.plot(YEAR4, FATALITIES1, label = TOP_FATALITIES.index[1])
plt.plot(YEAR5, FATALITIES2, label = TOP_FATALITIES.index[2])
plt.plot(YEAR6, FATALITIES3, label = TOP_FATALITIES.index[3])
plt.xlabel('Years')
plt.ylabel('Fatalities')
plt.title('Top 4 country of Fatalities')
plt.legend()
plt.tight_layout()
plt.savefig('plot.png')
plt.show()
```

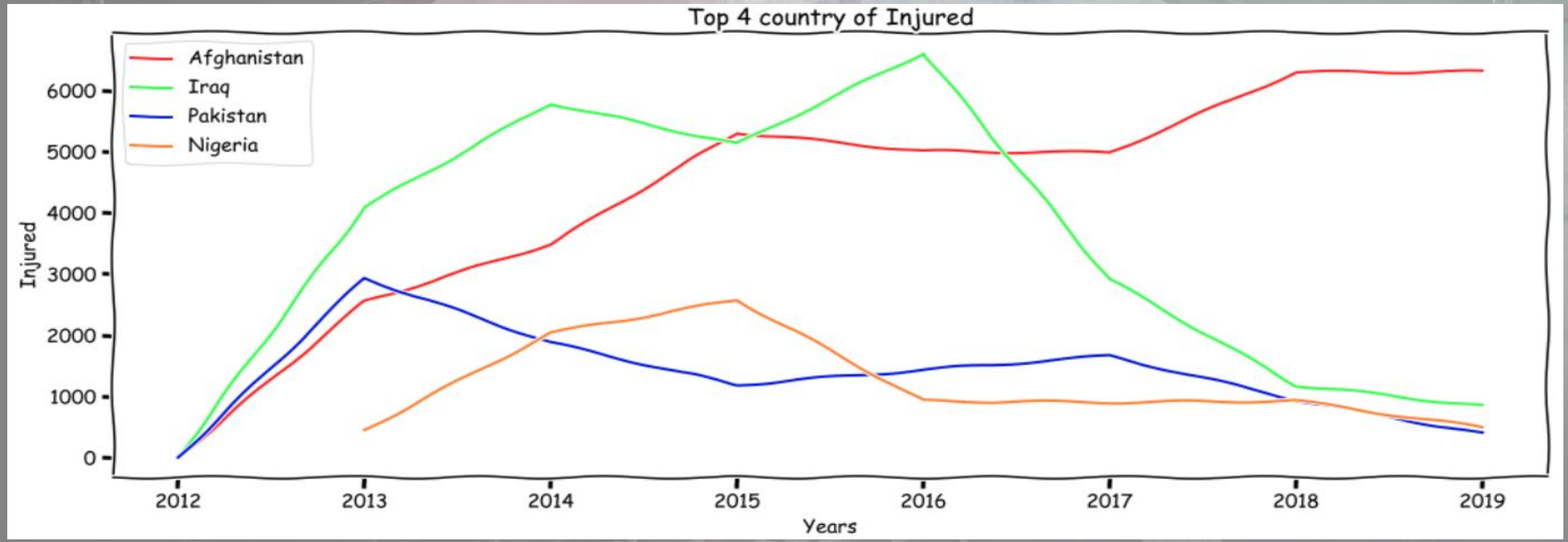


Question2:

Which four countries with most Injured?

One can learn from the graph who the four countries with the most wounded killed over the past decade are, as a direct result of the attacks that have taken place in their territory.

```
plt.xkcd()
plt.figure(figsize=[15,6])#size of the plt
plt.plot(YEAR0, INJURED0, color='#ff2121', label = TOP_INJURED.index[0])
plt.plot(YEAR1, INJURED1, color='#33FF4B', label = TOP_INJURED.index[1])
plt.plot(YEAR2, INJURED2, color='#0015df', label = TOP_INJURED.index[2])
plt.plot(YEAR33, INJURED33, color='#ff8133', label = TOP_INJURED.index[3])
plt.xlabel('Years')
plt.ylabel('Injured')
plt.title('Top 4 country of Injured')
plt.legend(loc='upper left')
plt.tight_layout()
plt.savefig('plot.png')
plt.show()
```

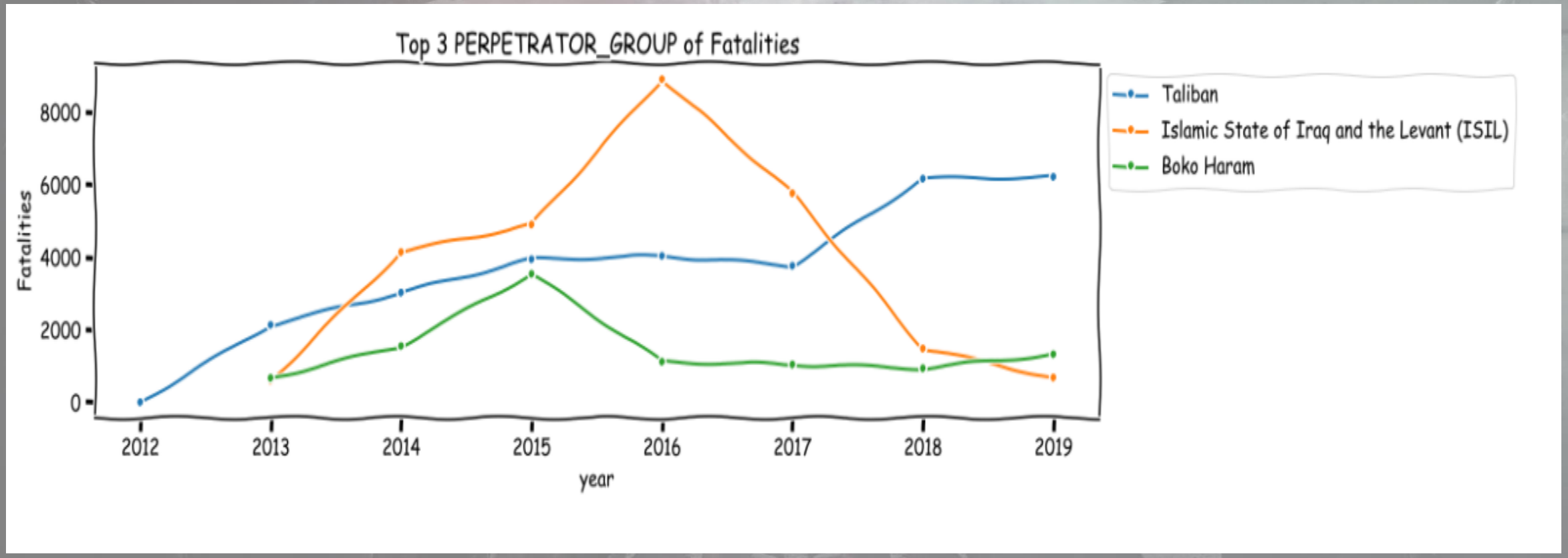


Question3:

Which three Terror organizations killed most human?

In the following graph we wanted to see the three organizations that caused the most deaths, as a result of terrorist attacks, all over the world.

```
plt.xkcd()
plt.figure(figsize=[15,4])#size of the plt
plt.plot(YEAR3, FATALITIES0, label = TOP_FATALITIES.index[0],marker='.' )
plt.plot(YEAR4, FATALITIES1, label = TOP_FATALITIES.index[1],marker='.'  )
plt.plot(YEAR5, FATALITIES2, label = TOP_FATALITIES.index[2],marker='.' )
plt.xlabel('year')
plt.ylabel('Fatalities')
plt.title('Top 3 PERPETRATOR_GROUP of Fatalities')
plt.legend(loc='upper left',bbox_to_anchor=(1,1))
plt.savefig('plot.png')
plt.show()
```

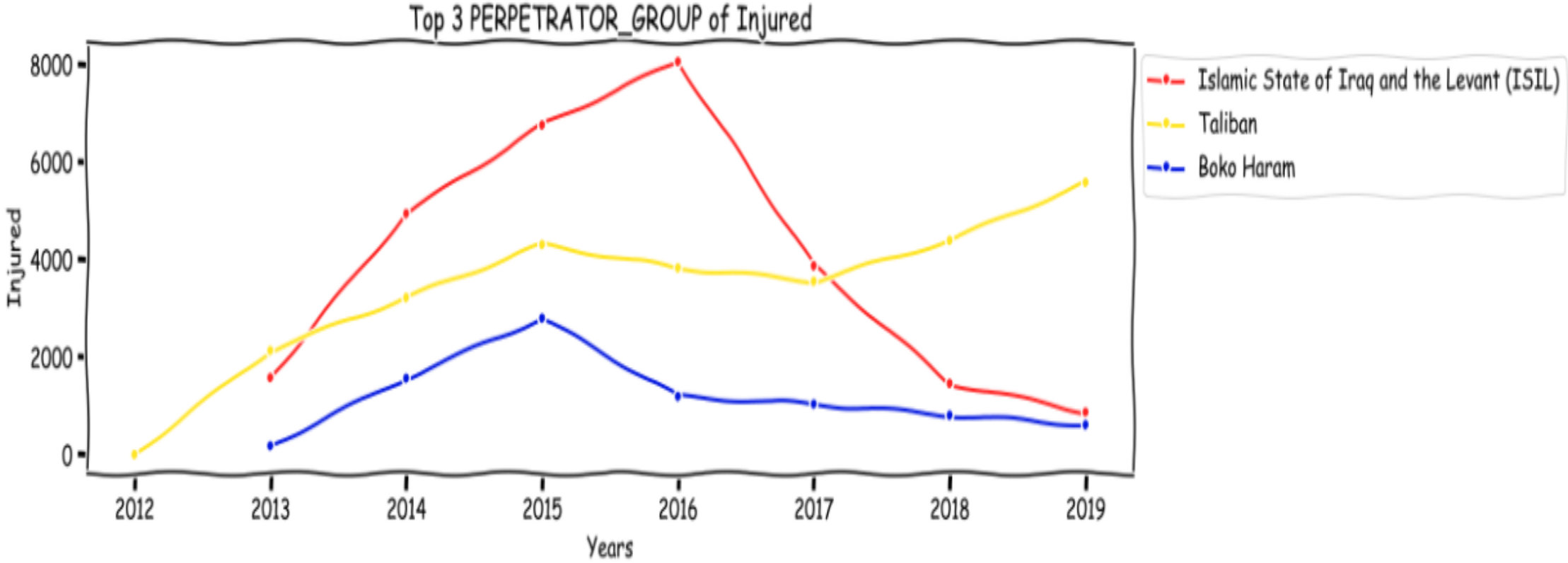


Question4:

Which three Terror organizations Injured most human?

We wanted to present in addition to the organizations that killed the most as a result of the attacks, also which terrorist organizations caused the most casualties around the world.

```
plt.xkcd()
plt.figure(figsize=[15,4])#size of the plt
plt.plot(YEAR0, INJURED0, color='#ff2121', label = TOP_INJURED.index[0],marker='.')
plt.plot(YEAR1, INJURED1, color='#ffe521', label = TOP_INJURED.index[1],marker='.')
plt.plot(YEAR2, INJURED2, color='#0015df', label = TOP_INJURED.index[2],marker='.')
plt.xlabel('Years')
plt.ylabel('Injured')
plt.title('Top 3 PERPETRATOR_GROUP of Injured')
plt.legend(loc='upper left',bbox_to_anchor=(1,1))
plt.savefig('plot.png')
plt.show()
```



Top Worst Terror Attacks in History

HeatMap

We wanted to see which attacks were most significant with the most casualties. The following graph shows the data in the configuration of a heat map when the color changes according to the number of people killed in the attack that took place.

```
import plotly.offline as py
py.init_notebook_mode(connected=True)
import plotly.graph_objs as go
colorscale = [[0, '#edf8fb'], [.3, '#00BFFF'],  [.6, '#8856a7'],  [1, '#810f7c']]
heatmap = go.Heatmap(z=all_data.FATALITIES, x=all_data.YEAR, y=all_data.COUNTRY, colorscale=col
data = [heatmap]
layout = go.Layout(
    title='Top  Worst Terror Attacks in History ',
    xaxis = dict(ticks='', ),
    yaxis = dict(ticks=''))
fig = go.Figure(data=data, layout=layout)
fig.update_layout(height=1000)
py.iplot(fig, filename='heatmap',show_link=False)
```

Prediction question machine learning stage

First model Fatalities by Year

In the first model, we used linear regression machine learning to predict the number of fatalities in a particular country.

```
By_Country_Year=all_data.groupby(['COUNTRY','YEAR']).sum()
```

```
By_Country_Year.reset_index()
By_Country_Year=By_Country_Year.groupby(['COUNTRY'])
```

```
dictreg={}
for c in By_Country_Year.groups:
    tr=By_Country_Year.get_group(c)
    tr=tr.reset_index().drop(columns=['COUNTRY'])
    a=tr['YEAR'].values.reshape(-1,1)
    v=tr['FATALITIES'].values.reshape(-1,1)
    m=linear_model.LinearRegression().fit(a,v)
    dictreg[c]=m
```

You can see how we put values in and allow regression to work and make future predictions.
Enter values manually by country and year, the prediction is made, and we are shown at the end the number of deaths
In the same country we wrote and in the desired year.

```
NameOfCountry = input("Please enter the country:")
Year = input("Please enter the Year:")
reg=dictreg[NameOfCountry].predict([[Year]])
print('By the regration:', reg ,'fatalities in', NameOfCountry ,'at' ,Year)
```

```
Please enter the country:Afghanistan
Please enter the Year:2022
By the regration: [[10189.83333333]] fatalities in Afghanistan at 2022
```

Please enter the country:

Please enter the Year:

Example

Build our Gradient Boosting regression model
[The second model]

```
# build our fist GradientBoosting regression model
dictreg={}
for c in By_Country_Month.groups:
    tr=By_Country_Month.get_group(c)
    tr=tr.reset_index().drop(columns=['COUNTRY'])
    a=tr['MONTH'].values.reshape(-1,1)
    v=tr['FATALITIES'].values
    m=ensemble.GradientBoostingRegressor().fit(a,v)
    dictreg[c]=m
```

Enter values manually by country and month, the prediction is made and we are shown at the end the number of deaths In the same country we wrote in a past average calculation of that month.

Please insert country and month to predict how many fatalities in a particular month

```
NameOfCountry = input("Please enter the country:")
Month = input("Please enter number of Month 1-12:    ")
reg=dictreg[NameOfCountry].predict([[Month]])
print('By the regration:', reg ,'fatalities in', NameOfCountry ,'at Month' ,Month )
```

```
Please enter the country:Iraq
Please enter number of Month 1-12:    5
By the regration: [287.04829983] fatalities in Iraq at Month 5
```

Please enter the country:

Please enter number of Month 1-12:

Example